Table of Contents

Security Conferences I have attended

Tell me about yourself

At present, I support small-medium customers through consulting - to migrate their workflows into AWS and create EC2 instances, pipelines, db and so for. I have also been involved in some ML/AI initiatives and hackathons that help me gain insights about new technologies related to ML/AI.

As a Senior Manager in Quality, at Demand Driven, I was focused on leading several teams and driving strategic vision, innovation and initiatives that resulted in enhancing the quality bar and automation across our test engineering department. I worked and collaborated with cross functional teams- business (atlanta, france), development (atlanta, redmond),test team (france, ukraine, costa-rica), Implementation team (atlanta, france) and global customer operations.

My primary contributions to the team were to:

(1) Understand the incoming feedback from our customers and capture the pain points and or areas of opportunity and derive mitigation plans that will translate into happy, satisfied customers.

(2) Understand our company business roadmap & goals, and how best aligned this with our strategic quality vision and test and automation initiatives
(3) Oversee all about our test activities and test process across the test engineering department and provide updates at different stages in software development cycle

A couple of success stories, I would like to share are:

1> Overhaul of the web portal, this was a data-driven push originating from our test department due to the high volume of bugs reported internally by our team and externally by our customers. Presented to the leadership team, there was an opportunity to enhance the way our software communicated with the customer, and ensure there were more banners and more accurate bubbles. In the end a full web portal overhaul was approved and after it was released to production we saw a big reduction of UI bugs | user experience bugs (15% of global queues), and help reassign our team members to new feature work or automation initiatives.

2> Proposed an API automation initiative oriented to remove repetitive, time-consuming, detailed work from the test team. Once buy-in was obtained, 300 APIs were automated and moved to a CI-CD pipeline which led to decreased regression testing for APIs from 3 to 1 days, boosting team productivity by 30%.

How do you define ROI?

ROI = BEFORE snapshot minus AFTER snapshot = should yield to resources, costs and time savings derived from the improvements applied to a process or software workflow or experience.
Snapshot
Reduced Costs (Resources Applied [Hours]) +
Reduced Costs (Resources SW | HW [licenses, monthly expenses, machines] +
Reduced Execution Time dedicated to a given task or set of tasks or workflow (experience) +
Improved quality bar | needle

KPI - Key Performance Indicators
Defect Detection Efficiency:
Measure the percentage of defects found before release versus those reported by customers post-release
Mean Time - inclusive of idle time – spent waiting
To Resolve, To Release

Customer Adoption Rate |
Customer Satisfaction Rate
Implementation success Rate
Quality Breakdown % -> Feature or User Experience or Workflow

OKR - Objectives & Key Results
Objective: Enhance Testing Efficiency
KR1: Reduce average testing cycle time by 20%.
KR2: Increase defect detection efficiency by 15%.

Objective: Improve Test Coverage and Quality
KR1: Achieve 90% test coverage for all major features
KR2: Reduce post-release defect rate by 25%.

Objective: Optimize Resource Utilization
KR1: Increase automated tests to 60% of global tests

or by customer groups (size) or geographical areas.
Know where you are today BASELINE and how could you improve? Path | Visio

Quality % be correlated and aligned to business goals and objectives

High Test Coverage Rate
Automated % vs Manual ->
Ratio of automated tests to manual Tests. Path | Vision

Testing Cycle Time -> Cycle Cost
Track time taken from the start to end of testing.
Aim to have shorter cycles that indicate efficient testing processes.

KR2: Reduce cost of testing by 10% without impacting quality

Objective: Foster a Culture of Continuous Improv

KR1: Implement quarterly training for the testing team on new methodologies and tools. Lessons learned. Improve documentations | onboarding |

KR2: Establish a feedback loop with the development team to enhance test effectiveness

Optimize to reduce costs without compromise quality

Error Budget Consumed? Cost in hours of downtime, or lost revenue or business productivity low >downtime

Bug Fix Time -> Incidents (reports by customers); Internal (reports by Quality or Business);
Average time taken to resolve reported issues. Faster resolutions can lead to higher product quality and customer satisfaction

Align KPIs and OKRs for Positive ROI

The trick to show the actual value delivered is understanding how to report the savings in the context of business workflows | user experiences.

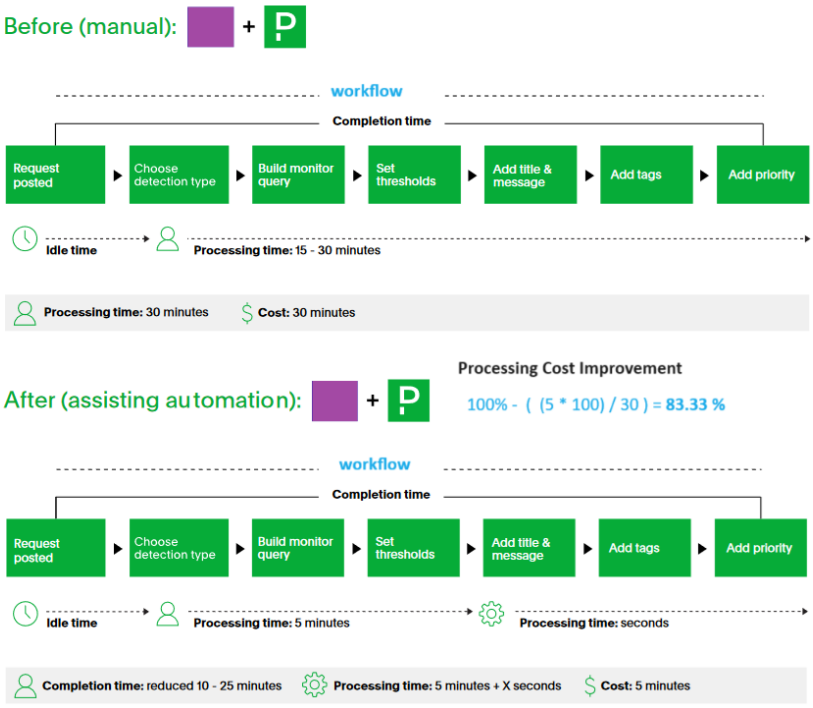
Process **Productivity**: Total completed cycles, or total benefit produced by a **workflow** over a given period of time.

Process **Efficiency**: Total completed cycles, or total benefit produced by a workflow **per person**, over a given period of time

Comparing Process Productivity and Process Efficiency values are a great way to show before and after effects of automation

- **Efficiency**: Faster testing and efficient resource use (automation) can reduce costs, contributing to ROI.
- **Quality**: High test coverage and defect detection rates lead to a more reliable product, enhancing customer satisfaction and potentially increasing sales. Promoting new features that are key to the company and add business value to customers.

- **Continuous Improvement**: Training and development and enhanced automation to ensure your team stays ahead of the curve, reducing the long-term costs of catching up and leveraging new technologies, trends and tools.



Can you share some of the latest responsibilities in your last role?

@Demand Driven

I was responsible for overseeing the entire testing strategy of the company across two geographical areas: US and Europe.
Oversee multiple, concurrent, and cross-dependency projects derived from our company's growing pains.

- ➔ Director of operations in France and Test team in France
- ➔ [direct report] Manager in Poland and QA Lead and Test Team in Ukraine {5}
- ➔ [direct report] Delivery Manager and QA Lead in Costa Rica + Team {6-8}
- ➔ Implementation Manager and UAT team in Atlanta
- ➔ Provide support (repro critical tickets) to Global Customer Success Operations Manager and team up on escalation and on-call rotation. (team spreaded in strategic locations)

I also had ownership of:

- ➔ Communicate with higher management and across leadership (business, development, customer support, implementation team) and key family and friends customers-partners
  - ◆ **testing vision, test strategy**, test roadmap + progress overtime, test insights & initiatives.
    *One of my primary tasks was to learn and accrue in-depth understanding of our company products, workflows, services, technologies applied, so later on, I can help translate our technical

workflows into clear, concise, simple to understand terms, that will enable our customers to gain insights quickly and as result, take informed actions and/or decisions.

- **present/talk Bi-Monthly or Quarterly** company meetings or workshops about Our test results, KPI , OKR, SLA metrics, mitigations/improvements focused towards customer support, showcase how customers were delighted with our new features, services and/or which pain points remain a challenge today and how we plan to mitigate them.

I was responsible of:

- **Keep up with understand latest trends**, new technologies, tools that can support better company goals, test strategies and meet customer demands
- **Identify future initiatives, features** (as result of analyze customer feedback) that will be aligned with our company business goals, but also help differentiate our company from other competitors and add value (i.e web portal overhaul with better user experiences, ensure the application communicated better with our cus
- **maximize resources utilization** (human <add, reallocate, remove>, handle budget vendors, training, online conferences, sw tools licenses, cloud usage, morale events, performance incentives | bonus, incentives applied to all company
- **call out risks, propose, define mitigations** - get feedback/buy-in as needed and oversee proper mitigations implementation

➔ Drive team growth, provide mentorship, guidance & visibility of career path as needed.

Oversee multiple, concurrent and complex projects - derived from company growing pains:

➔ Entire new portal UI overhaul
➔ Newly added (planned or unplanned small-big features) to a given release
➔ Strategic releases – with unique goals for improve performance | security | other
➔ Automation Initiatives + Continuous Improvement
➔ Sustainability support towards Customer Success team + on-call rotation
➔ Strategic Test Planning & Test Process, track progress of all testing activities
➔ Drive High Quality bar and implement best practices, ensure quality is embedded and can be evolved in software development
➔ Team development, growth, mentorship, performance reviews, perform hiring as needed

How do you prioritize tasks across projects with tight deadlines and multiple testing phases?

1st, I try to clearly **understand each project goal, its requirements and how success can be defined**

> what is the quality bar expected
> what is our ability & consistency to release on time, as well as, consider strategic releases
– what is the desired or intended release order?
– what are other plausible release scenarios, if X does not happen or meet the bar?
– what additional attributes should I consider: (1) geographical location | (2) demand | (3) high rate of adoption
– weight/impact of the features for our customers or as a key differentiator of our company from other competitors

2nd, I **evaluate the features with the highest risk and** ensure these are **prioritized 1st.** When talking about Risk, I consider - technical complexity, resource or time constraints, impact of features in our customers, dependencies between features or across projects or simply an estimated number of bugs expected to be found or added to our existing queues and our ability to process them & satisfy customer needs and & company business goals.

3rd, I **oversee and review with QA leadership, tasks & testing phases assignments to test team members based on their strengths and expertise**, to maximize efficiency and quality. (*) Leverage appropriate tools like JIRA, to create and see different pivots. Review appropriate resource capacity distribution (balanced work), high level review of top features estimations are not short or padded, vendor hours are capped and within our budget, account/plan for unexpected, resource vacations, holidays, training, etc. Number of tests expected to be executed in a given testing phase (in parallel or by best strategy), % of automation vs non-auto. Review ALL test dependencies are cleared prior to commence testing, enabling the test team for success. From time to time, based on project/team member (I have made strong recommendations or pre-assignments directly)

4th, **Leverage and promote automation** to allow our teams to focus on most complex or critical validations or to focus on recent/newly featured work not automated yet.

5th, **Share feedback as early as possible, with stakeholders or leadership team**.I promote a flow communication and share interim test results, helpful to gain insights
on how testing is progressing, review and remove any roadblocks as needed, confirm throughout check-points, that important milestones are: (1) on-track to be met [green], (2) at risk [yellow], if so call out as early as possible, current state and active mitigations in place or (3) off-track, raise the hand on which features might need to be deferred. Review, discuss and get buy-in of trade-offs prior to release timeframe. Leverage data-driven dashboards and automated results when/where is possible.

6th, **Remain flexible to adjust priorities as new information or changes in the project scope emerge**

7th, **Establish feedback loops** (full QA team, if possible include DEV too) and review what worked, what are areas of opportunity and derive proper mitigations, to promote a continuous improvement.

Tell me about yourself and why would you make a good senior manager?

Main focus is to drive the company forward and achieve strategic goals.
Motivate and inspire staff to believe in and work towards the company vision.

I've a track record for achieving company goals and delivering on targets as expected.
I consider myself a passionate leader where success has come as a result of putting the hours in, working smart and driving, motivating and inspiring my team towards the end goal.

What are your values as a senior manager?

Advocate for the highest quality standards and make data-driven decisions that balance the need for speed, for reliability and customer satisfaction.

1 - act with honesty and integrity in everything I do
2 - lead by example
3 - be accountable
4 - I look for continuous development and improvement, on both my staff and myself. It's important to be open to change and inspire and mentor the team to be the best they can be
5 - teamwork and great collaboration
6 - creativeness and innovation, encourage my team to be creative and bring new ideas that will benefit the organization
7 - improve and be obsessed with customer satisfaction

What qualities make a good leader?

Understand your role and impact on the organization, able to act and lead by example and be a positive role model, be able to communicate with staff and stakeholders, visualize goals and work hard to achieve them, recognize qualities in your staff and give them the opportunity to thrive, be adaptable to change at all times.

How do you build a positive relationship with your staff?

I've focused on 3 things: set standards, do recognition and be consistent

(1) Set the bar high and share with the team that we will focus on delivering with highest standards.
(2) Give them the opportunity to thrive in the role, and when they do: *recognize and reward* them. If staff/team feels rewarded and appreciated, they will likely work more effectively.
(3) All the staff needs to be treated consistently and fairly at all times.

With all above 3 things staff should feel empowered to achieve great things for the organization.

What are your long term goals?

I'm looking for stability and longevity within the role, company.
I want to focus on developing a strong culture and building a team around my area that are all empowered to achieve organizational objectives.
I want to continue to add to my education via certifications or masters.

Can you give an example of a challenging scalable testing problem you encountered and how you resolved it?

In demand driven, our supply chain customers (coca-cola, grief) were required to process (add) X-large jobs of data, to update their inventories and buffers thru our system, so accuracy was very critical. We were facing a significant challenge with performance testing, specifically related to load & scalability testing after refactoring baseline infrastructure components .

QA team initial tests showed that our supply chain could not handle the expected load having concurrently users Europe and USA (grief and coca-cola) together. The QA team was able to reproduce with a relatively ease that with a medium size number of jobs (data processing) , the system would become unresponsive, and response times were unacceptable. This was a major concern because our supply chain needed to support thousands of jobs of data processing concurrently across both US and Europe offices..

Identifying the Root Cause
The first step - identify the root cause of the performance problem. QA team engaged with Dev lead and dev seniors engineers to conduct an in-depth analysis of the application's architecture, refactoring, code, and database interactions.

Code Optimization
We found that some parts of the code were not optimized for handling large data sets efficiently. Additional caching mechanisms were needed in place. The QA team worked closely with the development team to refactor and optimize the code to improve data processing speed and reduce memory consumption.

Database Tuning
In addition, QA uncovered 2 critical bottlenecks in DB of our application's performance. We performed database profiling to identify slow-running queries and made necessary optimizations, such as adding indexes, rewriting queries, and configuring database parameters to handle high loads more efficiently.

Scaling
The QA team worked with the DevOps team to define an auto-scaling mechanism that could dynamically add more server instances during peak usage to ensure the system could handle the load without degradation in performance.

Performance Testing Iterations
We conducted multiple iterations of performance testing with increasingly larger loads to validate the improvements. This iterative approach allowed us to fine-tune the system's performance continuously.

In conjunction with the global customer operations team an alerting system to proactively identify and address any performance degradation in real-time was implemented.

Collaboration and Communication:
Effective communication and collaboration between the development, testing, and operations teams were crucial. We held regular cross-functional meetings to ensure everyone was aligned on the performance goals and progress.

Can you share an example where you implemented a security test strategy that significantly improved testing efficiency or quality?

Context: As a senior manager, I oversee the test engineering team for a large supply chain platform. Our challenge was to enhance the security of our platform without compromising the efficiency and speed of our testing processes.

Challenge: The platform had undergone a new web portal refactoring and as a result, there was a need to confirm no new vulnerabilities were injected, as well as, we needed to address vulnerabilities identified in the past, including known SQL injection and cross-site scripting (XSS) issues raised by our customers.

Automated Security Testing Integration:

- Implemented automated security testing tools within our CI/CD pipeline. This included tools like OWASP ZAP for detecting vulnerabilities and Snyk for scanning dependencies.
- Integrated these tools to run with every build, ensuring immediate feedback on new code.

Do Threat Modeling and Risk Assessment:

- Conducted regular threat modeling sessions with the development and operations teams to identify potential security threats.
- Prioritized testing based on risk assessment, focusing on critical areas or features such as 3rd party integrations and user data protection.

Security Testing Training for QA Teams:

- Provided specialized training for QA teams on security testing techniques and tools.
- Encouraged a 'security-first' mindset, integrating security considerations into all stages of testing.

Penetration Testing and Ethical Hacking:

- We conducted pen testing for high-critical business workflows and as well collaborated, received feedback from external security experts from our customers.
- Internal 'hackathons' were organized, encouraging teams to find and report vulnerabilities.

Performance and Stress Testing for Security:

- Included security aspects in performance testing, checking how the system behaved under stress or attack-like conditions.
- Ensured scalability of security measures under high traffic scenarios.

Outcomes:

- **Increased Detection and Prevention of Vulnerabilities:**
  The automated tools helped in early detection, significantly reducing the risk of security breaches.
- **Efficiency in Testing Process:**
  The integration of security testing into the CI/CD pipeline saved significant time, aligning testing with the fast-paced development environment.
- **Enhanced Team Capability:**
  Training and awareness raised the overall security posture of the team, with our QA engineers becoming strong and proactive in identifying security issues.
- **Resilience Against Attacks:**
  Regular penetration testing and security testing ensured the platform was well-prepared against potential attacks.

Embedding security into every stage of the testing lifecycle, we were able to maintain a robust defense against emerging threats and keep up with the rapid pace of development.

Can you describe a problem in which you had challenges and how did you approach to resolve it?
Problem: Inconsistent Test Environments

Scenario:
In demand driven, we faced a significant challenge with inconsistent test environments. The development team was working in an agile environment, releasing new features rapidly. However, the test environments were not updated synchronously with the production environment.
Unfortunately this led to a situation where the testing team was often working with inaccurate versions, or missing pieces of functionalty, causing discrepancies in test results and a lot of confusion.

Impact:

1. **False positives/negatives in testing**:
   Tests were passing/failing incorrectly due to environment mismatches.
2. **Delayed releases**:
   The inconsistencies often required retesting, which delayed the release of new features.
3. **Decreased confidence**:
   Testing team was concerned and started losing confidence in the testing process and results.

Resolution:

1. **Test - earlier in Feature Branches = and provide ahead as possible feedback.**
2. **Establishing an Environment Management Plan**:
   The first step was to develop a comprehensive environment management plan.
   This meant engaging in sessions with DevOps, Dev Team and QA team to outline guidelines for keeping the test environments in sync with the production environment, as well as understand how changes, upgrades and new features were going to be upgraded in the future.

   Automation was leveraged towards making deployments a more standardized process and better documented. This way everyone could better understand upcoming changes or look back at past changes. This helped as well, with environments being mirrored more accurately.

3. **Regular Environment Health Checks**:
   We introduced regular health checks for all test environments.
   These checks were automated to ensure the environments were operational and consistent with the production setup.
4. **Training and Awareness Sessions**:
   Conducted training sessions for the testing team to understand the importance of environment consistency and how to identify discrepancies.

5. **Feedback Loop**:
   Established a feedback loop where testers could report any inconsistencies or issues with the test environments promptly.Testers were able to rely on the results more confidently.

Can you describe a problem in which you had challenges and how did you approach to resolve it?
Problem: Vulnerability in user authentication

Context: During a routine security audit of our supply chain application, our testing team identified a critical vulnerability in the user authentication system after a single sign one feature was recently implemented and released. The issue was a flaw in the session management mechanism, which could potentially allow attackers to hijack user sessions.

Note: Session tokens are critical for ensuring that the user does not have to log in every time they make a new request to the server.

Resolution:

1. **Immediate Containment:**
   - Temporarily disabled the affected component to prevent any potential exploitation.
   - Notified the Global Customer support team and the relevant stakeholders about the issue.
2. **Root Cause Analysis:**
   - Conducted a thorough analysis to understand the root cause.
   - Found that the vulnerability was due to improper implementation of session tokens.
3. **Collaboration with Development Team:**
   - Worked closely with the development team to design a more robust session management system.
   - Ensured the implementation of best practices like secure token generation and management, and proper session timeout mechanisms.
4. **Implementing Fixes:**
   - Oversaw the development of the fix, ensuring minimal disruption to the service.
   - Conducted a code review to confirm the quality and security of the fix.
5. **Rigorous Testing:**
   - Performed a series of rigorous security tests, including penetration testing and vulnerability scanning, to ensure the issue was fully resolved.
   - Included tests for other potential security weaknesses in similar components.
6. **Documentation and Learning:**
   - Documented the incident, the steps taken to resolve it, and the lessons learned.
   - Conducted a session with the development and testing teams to share knowledge and prevent similar issues in the future.
7. **Long-Term Measures:**
   - Proposed internal “hackathons” and security audits and the establishment of a rapid response protocol for any future security issues.

Testing a payment gateway *involves several challenges* that need careful consideration:

1. **Security and Compliance**: Ensuring that the payment gateway complies with various security standards like PCI DSS (Payment Card Industry Data Security Standard) is crucial. It includes testing for data encryption, secure data storage, and protection against threats like hacking and fraud.
2. **Integration Testing**: Payment gateways must be integrated with various systems like websites, applications, and databases. Testing this integration to ensure seamless and error-free transactions is a challenge.
3. **User Interface Testing**: The payment process should be user-friendly. This involves testing the interface for ease of use, clarity of instructions, and error handling capabilities.
4. **Transaction Testing**: This includes testing for various scenarios like successful transactions, failed transactions, refunds, and payment disputes. Each scenario needs to be thoroughly tested to ensure the system behaves as expected.
5. **Performance Testing**: The payment gateway must handle high volumes of transactions without any delays or downtime. Performance testing under load and stress conditions is essential.
6. **Cross-Browser and Cross-Platform Testing**: Ensuring compatibility with different browsers and platforms is crucial since users will access the payment gateway through various devices and browsers.
7. **Handling Multiple Payment Methods**: Payment gateways often need to support multiple payment methods like credit/debit cards, digital wallets, and bank transfers. Testing each payment method for accuracy and reliability is necessary.
8. **Currency and Localization Testing**: For global operations, the gateway must handle multiple currencies and provide localized content. Testing for currency conversions and localization aspects is challenging.
9. **Security Updates and Patch Testing**: Regular updates and patches are released for security and functionality improvements. Testing these updates before deployment to ensure they don't introduce new issues is important.
10. **Compliance with Financial Regulations**: Payment gateways must adhere to various financial regulations of different countries. Testing for compliance with these ever-changing regulations can be complex.
11. **Error Handling and Recovery**: The system must handle errors gracefully and recover from failures without affecting the user experience or data integrity.
12. **Data Validation and Sanitization**: Testing for proper data validation and sanitization to prevent issues like SQL injection or cross-site scripting is vital.
13. **Monitoring and Reporting**: Continuous monitoring for fraud detection and generating accurate transaction reports is a challenge in testing.
14. **Accessibility Testing**: Ensuring that the payment gateway is accessible to users with disabilities is also an important aspect to consider.

How would you improve business efficiency and resilience with payment gateways?

1. **Enhanced Automation**:
   Automate repetitive tasks, regression testing, and where possible performance testing. Ensure that the payment gateway can handle various transaction loads without manual intervention.

2. **Build and Maintain a Secure Network and System:**
   Rely on a well defined firewall configuration to protect customer sensitive data and not using vendor-supplied defaults for system passwords and other security parameters. Maintain a Vulnerability Management Program that applies latest updates and patches, thru antivirus software, developing and maintaining secure systems and applications.

3. **Robust Security Testing**:
   Engage in regular vulnerability assessments**,** penetration testing and ensuring compliance with industry standards like PCI DSS.
   Implement Strong Access Control Measures, restricting access to customer sensitive data by business need-to-know, identify and authenticate access to system components, and restricting physical access to customer data.
   Overall, protect stored sensitive customer data and encrypting transmission of data across open, public networks.

4. **Performance Optimization**:
   Perform regular tests and performance optimizations of the payment gateway to handle high volumes of transactions smoothly during peak times.

5. **Disaster Recovery Plan**:
   Develop and regularly update a comprehensive disaster recovery plan.
   There is awareness and alignment in the event of a system failure, there is minimal disruption to services and quick recovery of operations.

6. **Cloud-Based Solutions and Scalability**:
   Leverage cloud technologies for better scalability and resilience. Cloud solutions can offer better uptime, distributed architecture for load balancing, and easier scalability during high-demand periods.

7. **Continuous Integration/Continuous Deployment (CI/CD) Practices**:
   Implementing CI/CD practices allows for frequent and reliable code deployments. This means that updates, bug fixes, and improvements can be rolled out quickly and

efficiently.

8. **Training and Development**:
   Regular training for the team on the latest technologies, testing methodologies, and industry trends is essential to keep the team's skills up-to-date and improve overall efficiency.

9. **Customer Feedback Loop**:
   Establish a strong feedback loop with customers to understand their needs and experiences. This can provide valuable insights for improving the efficiency and resilience of the payment gateway.